1 System Modelling

Continuous time (linear):

$$\dot{\boldsymbol{x}}(t) = \boldsymbol{A}\boldsymbol{x}(t) + \boldsymbol{B}\boldsymbol{u}(t), \boldsymbol{x}(0) = \boldsymbol{x}_0$$
$$\boldsymbol{y}(t) = \boldsymbol{C}\boldsymbol{x}(t) + \boldsymbol{D}\boldsymbol{u}(t)$$

Continuous time (nonlinear):

$$\dot{\boldsymbol{x}}(t) = \boldsymbol{f}(\boldsymbol{x}(t), \boldsymbol{u}(t)), \boldsymbol{x}(0) = \boldsymbol{x}_0$$
$$\boldsymbol{y}(t) = \boldsymbol{h}(\boldsymbol{x}(t), \boldsymbol{u}(t))$$

Discrete time (linear):

$$\boldsymbol{x}_{k+1} = \boldsymbol{A}_d\boldsymbol{x}_k + \boldsymbol{B}_d\boldsymbol{u}_k$$
$$\boldsymbol{y}_k = \boldsymbol{C}\boldsymbol{x}_k + \boldsymbol{D}\boldsymbol{u}_k$$

With a zero-order hold:

$$\begin{bmatrix} \boldsymbol{A}_d & \boldsymbol{B}_d \\ \boldsymbol{0} & \boldsymbol{I} \end{bmatrix} = \exp\left(\begin{bmatrix} \boldsymbol{A} & \boldsymbol{B} \\ \boldsymbol{0} & \boldsymbol{0} \end{bmatrix} h\right)$$

$$\boldsymbol{A}_d = e^{\boldsymbol{A}h}, \boldsymbol{B}_d = \int_0^h e^{\boldsymbol{A}\tau'} \mathrm{d}\tau' \boldsymbol{B}$$

2 Numerical Stability

Well conditioned problem: Cond_x small or $\mathrm{cond}_x \leq 1$

$$\mathrm{Cond}_x = \left|\frac{\Delta \bar{y}}{\Delta x}\right| \approx \left|\frac{\mathrm{d}f}{\mathrm{d}x}\right|$$

$$\mathrm{cond}_x = \left|\frac{\delta \bar{y}}{\delta x}\right| \approx |K_x| = \left|\frac{\mathrm{d}f}{\mathrm{d}x} \cdot \frac{x}{f(x)}\right|$$

φ is *order* p or $O(\Delta^p)$ if

$$\tilde{y} = \varphi(x, \Delta) - f(x) \propto \Delta^p$$

φ is *consistent* if

$$\lim_{\Delta \to 0} \varphi(x, \Delta) = f(x)$$

φ is *numerically stable* if

$$\left|\frac{\Delta \tilde{y}}{\Delta x}\right| \approx \left|\frac{\mathrm{d}\varphi}{\mathrm{d}x}\right| < 1$$

$\hat{y} = \varphi(\hat{x}, \Delta) = \varphi(\hat{x} + \Delta x)$ converges to $f(x)$ for $\Delta \to 0$ if it's consistent and at least marginally stable.

$\dot{\boldsymbol{x}} = \boldsymbol{A}\boldsymbol{x}$ well-conditioned/stable when $\mathrm{Re}(\lambda) < 0$ for all λ.

$\boldsymbol{x}_{k+1} = \boldsymbol{A}\boldsymbol{x}$ well-conditioned/stable when $|\lambda| < 1$ for all λ.

3 Numerical Root Finding

f is *Lipschitz continuous* if

$$\exists c \text{ s.t. } \forall x, z \in \mathbb{R}, |f(x) - f(z)| \leq c|x - z|$$

An algorithm has *convergence rate/order* r if

$$\lim_{k \to \infty} \frac{|E_{k+1}|}{|E_k|^r} = C$$

Bisection: order 1; requires continuity; have l_k, r_k such that $\operatorname{sgn} f(l_k) \neq \operatorname{sgn} f(r_k)$, check $c = \frac{l_k + r_k}{2}$ each iteration.

Fixed-point Iteration: order 2 if $g'(x^*) \approx 0$, 1 otherwise; requires update function $g(x^*) = x^*$ for Lipschitz g with $c < 1$ for $x \approx x^*$. To find $g(x)$, rearrange $f(x^*) = 0$ into the form $x^* = g(x^*)$. Start with guess, update as $x_{k+1} = g(x_k)$.

Newton's Method: order 2; requires continuous differentiability; equivalent to fixed-point iteration with $g(x) = x - \frac{f(x)}{f'(x)}$. Start with guess, update as $x_{k+1} = x_k - \frac{f(x_k)}{f'(x_k)}$. Exact for linear f and suffers for highly nonlinear f.

Secant Method: order between 1 and 2; Newton's method, but use $f'(x_k) \approx \frac{f(x_k) - f(x_{k-1})}{x_k - x_{k-1}}$.

4 Numerical Integration and Differentiation

Midpoint Rule: order 2, approx. f as constant between bounds

$$\int_{x_i}^{x_{i+1}} f(x)\,\mathrm{d}x \approx (x_{i+1} - x_i) \cdot f\left((x_{i+1} + x_i)/2\right)$$

Trapezoidal Rule: order 2, approx. f as linear

$$(x_{i+1} - x_i) \cdot \frac{f(x_{i+1}) + f(x_i)}{2}$$

Simpson's Rule: order 4, approx. f as quadratic

$$(x_{i+1} - x_i) \cdot \frac{f(x_{i+1}) + 4f\left((x_{i+1} + x_i)/2\right) + f(x_i)}{6}$$

Forward Difference: order 1

$$f'(x) \approx \frac{f(x + \Delta) - f(\Delta)}{\Delta}$$

Backward Difference: order 1

$$f'(x) \approx \frac{f(x) - f(x - \Delta)}{\Delta}$$

Centered Difference: order 2

$$f'(x) \approx \frac{f(x + \Delta) - f(x - \Delta)}{2\Delta}$$

5 Numerical ODE Solving

Forward Euler's Method: order 1 (global), 2 (local); explicit; conditionally stable if $|1 + h\lambda| < 1$.

$$x_{k+1} = x_k + hf_k$$

Backward Euler's Method: order 1 (global), 2 (local); implicit; conditionally stable if $|1 - h\lambda| > 1$.

$$x_{k+1} = x_k + hf_{k+1}$$

Trapezoidal Method: order 2 (global), 3 (local); implicit; conditionally stable if $\left|\frac{1 + \frac{1}{2}h\lambda}{1 - \frac{1}{2}h\lambda}\right| < 1$.

$$x_{k+1} = x_k + \frac{h}{2}(f_{k+1} + f_k)$$

Second-Order Runge-Kutta (Heun's Method, RK2): order 2 (global); explicit; conditionally stable if $-4 < 2h\lambda + h^2\lambda^2 < 0$.

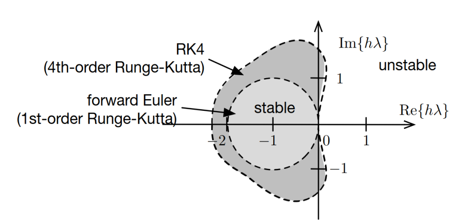
$$x_{k+1} = x_k + \frac{h}{2}\left(f(x_k + hf_k) + f_k\right)$$

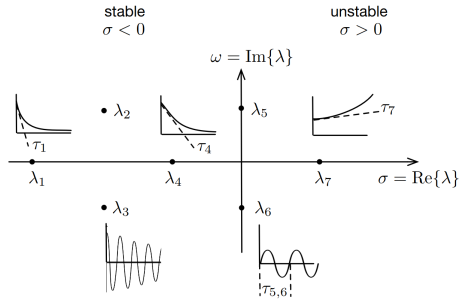
Fourth-Order Runge-Kutta (RK4): order 4 (global); explicit.

$$x_{k+1} = x_k + \frac{h}{6}(k_1 + 2k_2 + 2k_3 + k_4)$$

$$k_1 = f(x_k), \qquad k_2 = f\left(x_k + \frac{1}{2}hk_1\right)$$

$$k_3 = f\left(x_k + \frac{1}{2}hk_2\right), \qquad k_4 = f(x_k + hk_3)$$





Time constant: for $\dot{\boldsymbol{x}} = \boldsymbol{A}\boldsymbol{x}$ with complex eigenvalues λ_i:

$$\tau_i = \left\{ \frac{1}{|\mathrm{Re}(\lambda_i)|}, \frac{2\pi}{|\mathrm{Im}(\lambda_i)|} \right\}$$

This ODE is *stiff* if

$$\gamma = \frac{\tau_{\max}}{\tau_{\min}} > 10^3$$

Rule of thumb: simulate $T = 5\tau_{\max}$ if system is stable; use step size $h = \min\left\{ \frac{\tau_{\min}}{10}, \frac{T}{200} \right\}$ or number of steps $k = \max\{ 50\gamma, 200 \}$; plot with step size $H = \frac{T}{200} = \frac{\tau_{\max}}{40}$; for stiff systems use a variable-step solver with initial step $\frac{\tau_{\min}}{10}$. Use adaptive step size solvers for stiff problems.

6 Optimization

Formulation: minimize $f(\boldsymbol{x})$, subject to $\boldsymbol{g}(\boldsymbol{x}) = \boldsymbol{0}$ (equality constraints) and $\boldsymbol{h}(\boldsymbol{x}) \geq \boldsymbol{0}$ (inequality constraints), where $f : \mathbb{R}^n \mapsto \mathbb{R}, \boldsymbol{g} : \mathbb{R}^n \mapsto \mathbb{R}^m, \boldsymbol{h} : \mathbb{R}^n \mapsto \mathbb{R}^p$.

$\boldsymbol{x}^*$ is a *global minimum* if

$$\forall \boldsymbol{x} \in \mathbb{R}^n, f(\boldsymbol{x}^*) \leq f(\boldsymbol{x})$$

$\boldsymbol{x}^*$ is a *local minimum* if

$$\exists \varepsilon > 0 \text{ s.t. } \forall \boldsymbol{x} \in \mathbb{R}^n, \|\boldsymbol{x} - \boldsymbol{x}^*\|_2 < \varepsilon \implies f(\boldsymbol{x}^*) \leq f(\boldsymbol{x})$$

All minima of f are *stationary points*, where $\vec{\nabla} f(\boldsymbol{x}) = \boldsymbol{0}$; to find the global minimum, check all stationary points and boundaries.

$$\boldsymbol{H}_f(\boldsymbol{x}) = \begin{bmatrix} \frac{\partial^2 f}{\partial x_1 \partial x_1} & \cdots & \frac{\partial^2 f}{\partial x_1 \partial x_n} \\ \vdots & \ddots & \vdots \\ \frac{\partial^2 f}{\partial x_n \partial x_1} & \cdots & \frac{\partial^2 f}{\partial x_n \partial x_n} \end{bmatrix}$$

If $\boldsymbol{x}^*$ is a stationary point, it is a local minimum if $\boldsymbol{H}_f(\boldsymbol{x}^*) > 0$, maximum if $\boldsymbol{H}_f < 0$, saddle if $\boldsymbol{H}_f = 0$.

A *feasible point* is any $\boldsymbol{x}$ satisfying all constraints; the *feasible set* contains all feasible points. A *critical point* is a local maximum, minimum or saddle point in the feasible set.

Lagrange Multipliers: for equality constraints $\boldsymbol{g}(\boldsymbol{x}) = \boldsymbol{0}$:

$$\min_{\boldsymbol{x}, \boldsymbol{\lambda}} \Lambda(\boldsymbol{x}, \boldsymbol{\lambda}) = f(\boldsymbol{x}) - \boldsymbol{\lambda}^T \boldsymbol{g}(\boldsymbol{x})$$

Karush-Kuhn-Tucker (KTT) Conditions: $\boldsymbol{x}^*$ is a critical point when there exists $\boldsymbol{\lambda} \in \mathbb{R}^m$ and $\boldsymbol{\mu} \in \mathbb{R}^p$ such that:

1. Stationarity: $\vec{\nabla} f(\boldsymbol{x}^*) - \sum_i \lambda_i \vec{\nabla} g_i(\boldsymbol{x}^*) - \sum_j \mu_j \vec{\nabla} h_j(\boldsymbol{x}^*) = \boldsymbol{0}$
2. Primal feasibility: $\boldsymbol{g}(\boldsymbol{x}^*) = \boldsymbol{0}$ and $\boldsymbol{h}(\boldsymbol{x}^*) \geq \boldsymbol{0}$
3. Complementary slackness: $\forall j, \mu_j h_j(\boldsymbol{x}^*) = 0$
4. Dual feasibility: $\forall j, \mu_j \geq 0$

f is *convex* if $\boldsymbol{H}_f > 0$ for all $\boldsymbol{x}$, or
$\forall \boldsymbol{x}_1 \neq \boldsymbol{x}_2, \forall \alpha \in (0,1), f((1-\alpha)\boldsymbol{x}_1 + \alpha \boldsymbol{x}_2) \leq (1-\alpha) f(\boldsymbol{x}_1) + \alpha f(\boldsymbol{x}_2)$

A set S is *convex* if:

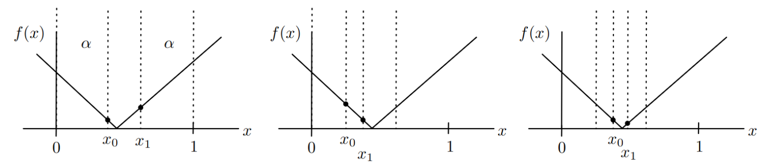
$$\forall \boldsymbol{x}, \boldsymbol{y} \in S, \alpha \in [0,1], \alpha \boldsymbol{x} + (1-\alpha)\boldsymbol{y} \in S$$

If the objective f and feasible set are both convex, the problem is convex and has a unique minimum.

7 Numerical Optimization Algorithms

f is *unimodular* over $[a,b]$ if there exists $x^* \in [a,b]$ s.t. f is decreasing over $[a, x^*]$ and increasing over $[x^*, b]$.

Golden Section Search: 1D, unconstrained; order 1; requires unimodularity. Rescale to $[0,1]$, choose $x_0 = \alpha, x_1 = 1 - \alpha$ for $0 < \alpha < 1/2$; evaluate $f(x_0), f(x_1)$, discard larger side, rescale and repeat. Using $1 - \alpha = \frac{1}{2}(\sqrt{5} - 1)$ allows using x_0 as the next x_1.



Gradient Descent: unconstrained; requires twice differentiable. Let $g(\alpha) = f(\boldsymbol{x}_k - \alpha \vec{\nabla} f(\boldsymbol{x}_k)^T)$, find α^* minimizing g, update $x_{k+1} = x_k - a^* \vec{\nabla} f(\boldsymbol{x}_k)^T$. Optimizing g is expensive so fixed step size can be used. Suffers from poor conditioning of f.

Newton's Method: unconstrained; requires twice differentiable. Iterate as $\boldsymbol{x}_{k+1} = \boldsymbol{x}_k - \boldsymbol{H}_f^{-1}(\boldsymbol{x}_k) \vec{\nabla} f(\boldsymbol{x}_k)^T$; or in 1D, $x_{k+1} = x_k - \frac{f'(x_k)}{f''(x_k)}$. Gauss-Newton approximates $\boldsymbol{H}_f$ using first derivatives (mix of gradient descent and Newton). Levenberg-Marquardt applies adaptive regularization for nearly-singular Hessians.

Sequential Quadratic Programming (SQP): constrained. Iteratively solves a series of simpler, less constrained approximations of the problem. f is replaced by a quadratic approximation and the constraints linearized. Similar to Newton's method; only converges if initial guess is good.

Barrier Methods: constrained. Constraints are turned into penalties on the objective. Define new objective as $f'(x) = f(x) + \rho \frac{1}{h(x)}$ where weight ρ is increased to satisfy constraints, decreased for more accuracy.

8 Linear Algebra

$\boldsymbol{A}^T \boldsymbol{A}$ is positive definite if $\boldsymbol{A}$ is full rank, semi-definite otherwise.
Orthogonal $\boldsymbol{Q}$ rotates, preserves angles, and $\boldsymbol{Q}^T \boldsymbol{Q} = \boldsymbol{1}$.
$\boldsymbol{A}\boldsymbol{x} = \boldsymbol{b}$ can be solved by factoring $\boldsymbol{A} = \boldsymbol{L}\boldsymbol{U}$ and solving $\boldsymbol{L}\boldsymbol{y} = \boldsymbol{b}, \boldsymbol{U}\boldsymbol{x} = \boldsymbol{y}$.
A general *vector norm* satisfies: $\|\boldsymbol{x}\| = 0 \iff \boldsymbol{x} = \boldsymbol{0}$; $\|c\boldsymbol{x}\| = |c|\|\boldsymbol{x}\|$; $\|\boldsymbol{x} + \boldsymbol{y}\| \leq \|\boldsymbol{x}\| + \|\boldsymbol{y}\|$. The *p-norm* for $p \geq 1$ is convex:

$$\|\boldsymbol{x}\|_p = (|x_1|^p + |x_2|^p + \cdots + |x_n|^p)^{\frac{1}{p}}$$

2-norm is Euclidean; ∞-norm is $\|\boldsymbol{x}\|_\infty = \max(|x_1|, |x_2|, \ldots, |x_n|)$.
The *matrix norm* induced by a vector norm is

$$\|\boldsymbol{A}\| = \max\{ \|\boldsymbol{A}\boldsymbol{x}\| \mid \|\boldsymbol{x}\| = 1 \} = \max_{\boldsymbol{x} \in \mathbb{R}^n, \boldsymbol{x} \neq 0} \frac{\|\boldsymbol{A}\boldsymbol{x}\|}{\|\boldsymbol{x}\|}$$

1-norm: maximum column sum

$$\|\boldsymbol{A}\|_1 = \max_{1 \leq j \leq n} \sum_{i=1}^{m} |a_{ij}|$$

2-norm: largest eigenvalue of $\boldsymbol{A}$ (aka *spectral radius*)

$$\|\boldsymbol{A}\|_2 = \max\left\{ \sqrt{\lambda} \;\middle|\; \exists \boldsymbol{x} \in \mathbb{R}^n \text{ s.t. } \boldsymbol{A}^T \boldsymbol{A}\boldsymbol{x} = \lambda \boldsymbol{x} \right\}$$

Frobenius norm: always $\|\boldsymbol{A}\|_2 \leq \|\boldsymbol{A}\|_F$

$$\|\boldsymbol{A}\|_F = \sqrt{\sum_{i=1}^{m} \sum_{j=1}^{n} |a_{ij}|^2} = \sqrt{\operatorname{tr} \boldsymbol{A}^T \boldsymbol{A}}$$

∞-norm: maximum row sum

$$\|\boldsymbol{A}\|_\infty = \max_{1 \leq i \leq m} \sum_{j=1}^{n} |a_{ij}|$$

The *condition number* of $\boldsymbol{A}$ with respect to a norm is

$$\operatorname{cond} \boldsymbol{A} = \|\boldsymbol{A}\| \|\boldsymbol{A}^{-1}\|$$

or ∞ if $\boldsymbol{A}$ is non-invertible. For solving $\boldsymbol{A}\boldsymbol{x} = \boldsymbol{b}$, this describes how error in $\boldsymbol{A}$ and $\boldsymbol{b}$ propagates to $\boldsymbol{x}$.